KEYD: SECURE KEYDEDUPLICATION WITH IDENTITY-BASED BROADCAST ENCRYPTION

[1]K. Chetan Kumar, [2]Dr. N. Chandramouli
[1]M.Tech, Department of Computer Science, VAAGESWARI COLLEGE OF ENGINEERING Thimmapur, Karimnagar, Telangana, INDIA, H.NO: 20S41D5825, chetankonda1991@gmail.com
[2]HOD of CSE, Department of Computer Science, VAAGESWARI COLLEGE OF ENGINEERING Thimmapur, Karimnagar, Telangana, INDIA, vgse.csehod@gmail.com

ABSTRACT: Deduplication, which can save storage cost by enabling us to store only one copy of identical data, becomes unprecedentedly significant with the dramatic increase in data stored in the cloud. For the purpose of ensuring data confidentiality, they are usually encrypted before outsourced. Traditional encryption will inevitably result in multiple different ciphertexts produced from the same plaintext by different users' secret keys, which hinders data deduplication. Convergent encryption makes deduplication possible since it naturally encrypts the same plaintexts into the same ciphertexts. One attendant problem is how to reliably and effectively manage a huge number of convergent keys. Several deduplication schemes have been proposed to deal with the convergent key management problem. However, they either need to introduce key management servers or require interaction between data owners. In this paper, we design a novel client-side deduplication protocol named KeyD without such an independent key management server by utilizing the identity-based broadcast encryption (IBBE) technique. Users only interact with the cloud service provider (CSP) during the process of data upload and download. Security analysis demonstrates that KeyD ensures data confidentiality and convergent key security, and well protects the ownership privacy simultaneously. A thorough and detailed performance comparison shows that our scheme makes a better tradeoff among the storage cost, communication and computation overhead.

Keywords *–Data deduplication, convergent encryption, convergent key management, identity-based broadcast encryption.*

1. INTRODUCTION

The stored data is growing intensely with the advent of the era of Big Data. We need to constantly increase the storage devices if we continue using the traditional storage way. Alternatively, more and more users are prone to outsource their storage to cloud such as Amazon Web Services (AWS) [1] for economic savings. The ever-increasing data and users, coupled with multiple backup and other factors, result in more and more duplication of files or blocks in the cloud. In order to improve the storage efficiency in the pay-as-yougo model [2], deduplication operation is adopted for eliminating duplicate copies of redundant data on the cloud-side. Consider an example that m users outsource the same data copies1 of n TB to the CSP. With data deduplication, only one copy is actually stored in the cloud, and the subsequent instances are referenced back to the saved copy for reducing storage roughly from mn to n TB. However, in order to protect the safety of the outsourced data, they are usually encrypted by their owners before outsourced to the CSP. Then it comes the problem, how can the CSP perform deduplication when these same data copies are encrypted into different ciphertexts by different users? Convergent encryption (CE) [3], which encrypts a data copy with a convergent key derived by computing the cryptographic hash value of the content of the data copy itself and thus can produce identical ciphertext from identical plaintext, brings the hope to realize deduplication while ensuring data confidentiality. This property of convergent encryption allows the CSP to perform deduplication on encrypted data.

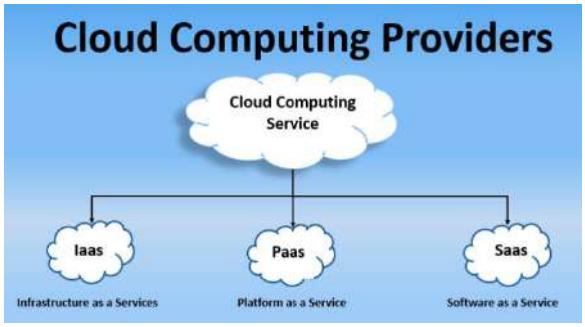

Fig.1: Example figure

In particular, users encrypt their data copies using corresponding convergent keys and outsource encrypted data to the CSP. They just need to keep convergent keys locally so that they can later restore the data. However, the number of convergent keys increases linearly with the number of data copies since a data copy corresponds to a convergent key. As we all know, in practical file storage systems such as Google File System GFS [4] and Hadoop Distribute File System HDFS [5], data files are usually divided into fine-grained blocks to facilitate deduplication management, which makes the convergent key storage even more serious. Suppose the n TB data in the above example is divided into blocks of size 4 KB each, and that each convergent key is the hash value of SHA-256. Then each owner has to store a total size of 8n GB2 of convergent keys. Such a storage overhead is still a great burden for the user. A naive solution to this problem is described in [6]. Each data owner encrypts all his data copies using the corresponding convergent keys and further encrypts these convergent keys using his master key. Both encrypted data copies andconvergent keys are stored in the cloud, and users keep their master keys and the metadata about the outsourced data locally. Although the encrypted data can be deduplicated by the cloud, the storage of encrypted convergent keys will increase linearly with the number of users. Performing deduplication on these encrypted convergent keys is just as important as performing deduplication on encrypted data copies.

2. LITERATURE REVIEW

Reclaiming Space from Duplicate Files in a Serverless Distributed File System

The Farsite distributed file system provides availability by replicating each file onto multiple desktop computers. Since this replication consumes significant storage space, it is important to reclaim used space where possible. Measurement of over 500 desktop file systems shows that nearly half of all consumed space is occupied by duplicate files. We present a mechanism to reclaim space from this incidental duplication to make it available for controlled file replication. Our mechanism includes: (1) convergent encryption, which enables duplicate files to be coalesced into the space of a single file, even if the files are encrypted with different users' keys; and (2) SALAD, a Self-Arranging Lossy Associative Database for aggregating file content and location information in a decentralized, scalable, fault-tolerant manner. Large-scale simulation experiments show that the duplicate-file coalescing system is scalable, highly effective, and fault-tolerant.

Secure Deduplication with Efficient and Reliable Convergent Key Management

Data deduplication is a technique for eliminating duplicate copies of data, and has been widely used in cloud storage to reduce storage space and upload bandwidth. Promising as it is, an arising challenge is to perform secure deduplication in cloud storage. Although convergent encryption has been extensively adopted for secure deduplication, a critical issue of making convergent encryption practical is to efficiently and reliably manage a huge number of convergent keys. This paper makes the first attempt to formally address the problem of achieving efficient and reliable key management in secure deduplication. We first introduce a baseline approach in which each user holds an independent master key for encrypting the convergent keys and outsourcing them to the cloud. However, such a baseline key management scheme generates an enormous number of keys with the increasing number of users and requires users to dedicatedly protect the master keys. To this end, we propose Dekey , a new

construction in which users do not need to manage any keys on their own but instead securely distribute the convergent key shares across multiple servers. Security analysis demonstrates that Dekey is secure in terms of the definitions specified in the proposed security model. As a proof of concept, we implement Dekey using the Ramp secret sharing scheme and demonstrate that Dekey incurs limited overhead in realistic environments.

Multiple Ramp Schemes

A (t,k,n,S) ramp scheme is a protocol to distribute a secret s chosen in S among a set P of n participants in such a way that: (1) sets of participants of cardinality greater than or equal to k can reconstruct the secret s; (2) sets of participants of cardinality less than or equal to t have no information on s, whereas (3) sets of participants of cardinality greater than t and less than k might have "some" information on s. In this correspondence we analyze multiple ramp schemes, which are protocols to share many secrets among a set P of participants, using different ramp schemes. In particular, we prove a tight lower bound on the size of the shares held by each participant and on the dealer's randomness in multiple ramp schemes.

Secure Data Deduplication with Reliable Key Management for Dynamic Updates in Cpss

With the increasing sensing and communication in cyber physical social system (CPSS), the data volume is growing much rapidly in recent years. Secure deduplication has attracted considerable interests of storage provider for data management efficiency and data privacy preserving. One of the most challenging issues in secure deduplication is how to manage data and the convergent key when users frequently update it. To solve this problem, D. Koo et al. use bilinear paring as the key method. However, bilinear paring requires high computation cost for implementations. In this paper, we propose a session-key-based convergent key management scheme, named SKC, to secure the dynamic update in the data deduplication. Specifically, each data owner in SKC can verify the correctness of the session key and dynamically change it with the data update. Furthermore, to enable group combination and remove the aid of gateway (GW), a convergent key sharing scheme, named CKS, is presented. Security analysis demonstrates that both SKC and CKS can protect the confidentiality of the data and the convergent key in the case of dynamic updates. The simulation results show that our SKC and CKS can significantly reduce computation complexity and communication during the data uploading phase.

Dynamic Data Deduplication in Cloud Storage

Cloud computing plays a major role in the business domain today as computing resources are delivered as a utility on demand to customers over the Internet. Cloud storage is one of the services provided in cloud computing which has been increasing in popularity. The main advantage of using cloud storage from the customers' point of view is that customers can reduce their expenditure in purchasing and maintaining storage infrastructure while only paying for the amount of storage requested, which can be scaled-up and down upon demand. With the growing data size of cloud computing, a reduction in data volumes could help providers reducing the costs of running large storage system and saving energy consumption. So data deduplication techniques have been brought to improve storage efficiency in cloud storages. With the dynamic nature of data in cloud storage, data usage in cloud changes overtime, some data chunks may be read frequently in period of time, but may not be used in another time period. Some datasets may be frequently accessed or updated by multiple users at the same time, while others may need the high level of redundancy for reliability requirement. Therefore, it is crucial to support this dynamic feature in cloud storage. However current approaches are mostly focused on static scheme, which limits their full applicability in dynamic characteristic of data in cloud storage. In this paper, we propose a dynamic deduplication scheme for cloud storage, which aiming to improve storage efficiency and maintaining redundancy for fault tolerance.

3. METHODOLOGY

Li et al. employs the Ramp secret sharing scheme (RSSS) to construct secret shares for the convergent keys and distribute them across multiple independent Key Management Cloud Service Providers (KM-CSPs). To recover data copies, a user must access a minimum number of key servers through authentication and obtain the secret shares to reconstruct the convergent keys. The premise of the scheme security is that the number of colluded KM-

CSPs is not more than a predefined threshold, such that a convergent key cannot be guessed by the colluded KM-CSPs.

- Wen et al. propose a session-key-based convergent key management scheme and an improved version. A trustable gateway is needed to check the duplication and buffer the new data blocks for periodically uploading them to the cloud storage server (CSS). As far as we know, there is no deduplication scheme that can effectively manage convergent keys, without the aid of additional independent key management servers or other trusted parties like Gateway.

Disadvantages:

- Storing convergent keys locally is quite a heavy burden for users, but encrypting them with different master keys and outsourcing them will lead to the duplication of encrypted convergent keys on the server side.
- Involves complex interactions between clients for key sharing, which violates the ownership privacy during the data deduplication. Therefore, designing a secure and effective client-side deduplication scheme without other independent servers or trusted third parties is still an open problem.

In this paper, we construct a secure deduplication scheme to manage convergent keys.

- We propose a novel client-side deduplication scheme. Specifically, we make a combination of convergent encryption (CE) and ID-based broadcast encryption (IBBE) to achieve secure and efficient convergent key management, without introducing any other independent key management servers or trusted third parties.
- Security analysis demonstrates that our scheme ensures the confidentiality of data files and the security of convergent keys.

Advantages:

- A comprehensive performance comparison between KeyD and several present works is given, showing that our scheme makes a better tradeoff among the storage cost, communication overhead and computation overhead.
- Fully protected against outside attackers and the CSP.
- No Additional Independent Servers or Trusted Third Parties.
- Confidentiality of Data.
- Semantic Security of Convergent Keys.

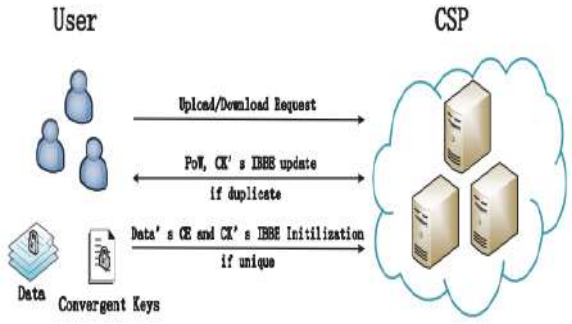

Fig.2: System architecture

We adopt the Identity-Based Broadcast Encryption to encrypt convergent keys before sending them to the Cloud Service Provider (CSP). A necessary authority involved in an IBBE is the Private Key Generator PKG. Using its master secret key MSK, the PKG can generate a decryption key skIDi for each new member with identity IDi to decrypt messages. An attractive feature of the IBBE scheme is that the broadcaster does not hold any private

4. IMPLEMENTATION

MODULES:

- User
- Cloud Service Provider (CSP)

MODULES DESCSRIPTION:

User:

The *user* owns a large amount of data, and he is usually limited in his storage space and computing capability. So he wants to outsource the data to the cloud and access them later. For the purpose of saving network bandwidth in the data upload phase, the user only uploads the data that has not been stored in the cloud. If duplication happens, he runs the PoW protocol with the CSP to prove his ownership, and updates materials related to the convergent key (CK)'s identity-based broadcast encryption (IBBE) with the help of CSP. Otherwise, he CE-encrypts the data and IBBE-encrypts the convergent key respectively and then uploads them. In our setting, the user only needs to protect his secret key and maintain his ID so that he can later decrypt the convergent keys to further retrieve the outsourced blocks.

Cloud Service Provider (CSP):

The *CSP* is an entity that provides cloud storage services, including data, convergent keys and other relative materials. It maintains a list of tags of data files and blocks for checking deduplication, and helps the user to do some expensive computations. When the CSP receives a file upload request, it first checks if the file duplicates. If yes, it informs the user to prove his ownership of the file, returns the file pointer to the user, and updates the materials related to CKs' IBBE version of all the blocks making up this file.

5. EXPERIMENTAL RESULTS

Fig.3: Index page

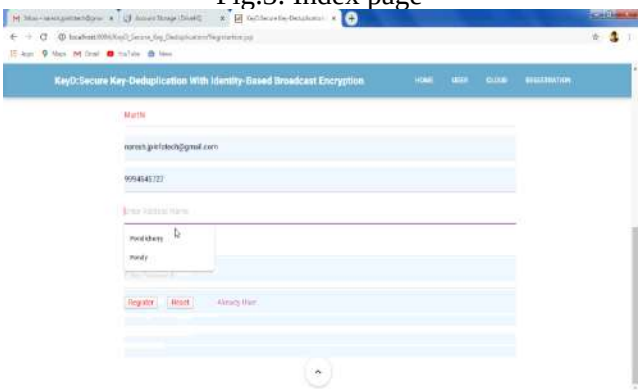

Fig.4: Register page

ISSN: 0972-2750
Vol-12 Issue-11 No.01: 2022

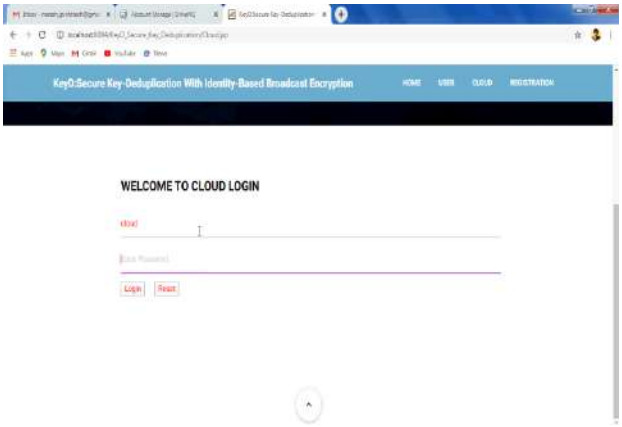

Fig.5: Cloudlogin.jsp

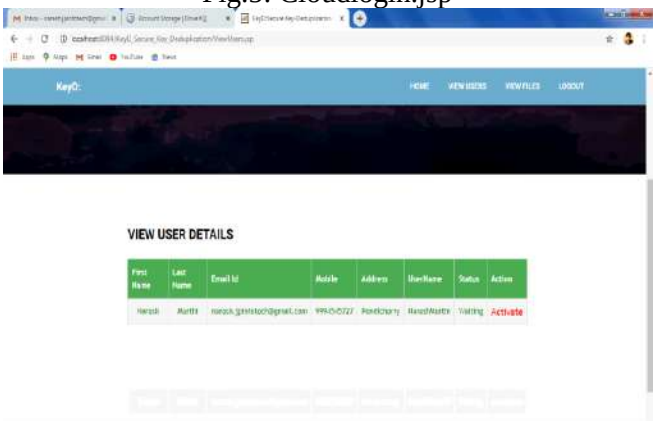

Fig.6: View Users

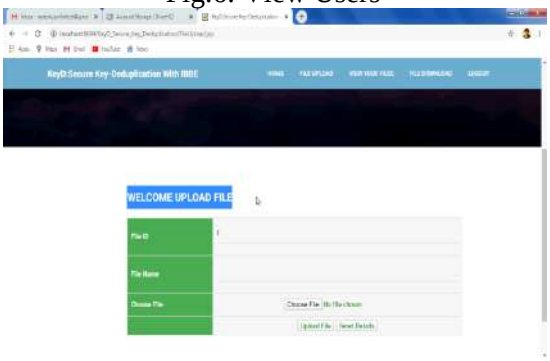

Fig.7: FileUpload

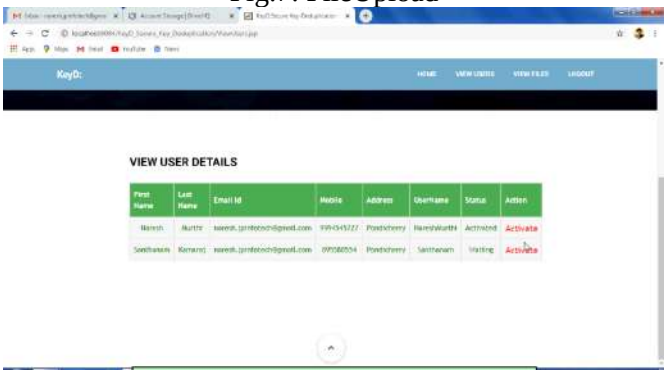

Fig.8: View User

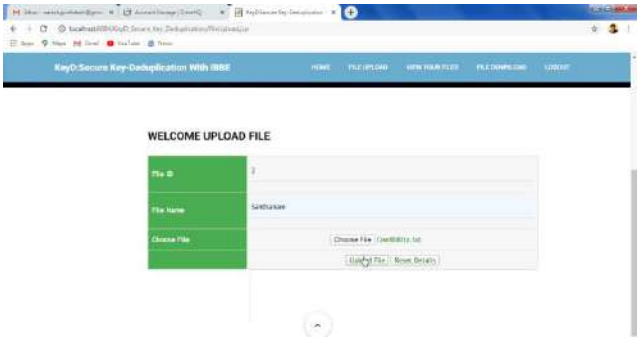

Fig.9: Fileupload

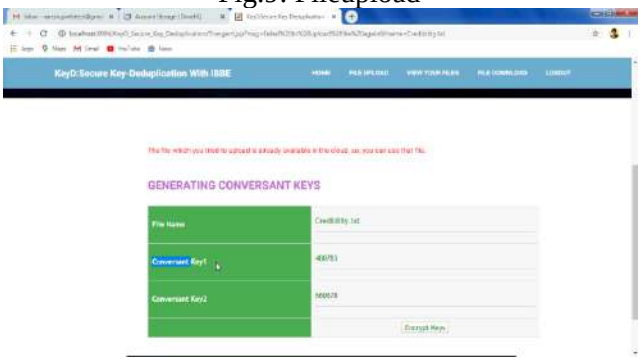

Fig.10: Trangent

Fig.11: EncriptKeys

Fig.12: GetKeys

6. CONCLUSION

In this paper, we propose a secure client-side deduplicationscheme KeyD to effectively manage convergent keys. Datadeduplication in our design is achieved by interactionsbetween data owners and the Cloud Service Provider

(CSP),without participation of other trusted third parties or KeyManagement Cloud Service Providers. The security analysisshows that our KeyD ensures the confidentiality of dataand security of convergent keys, and well protects the userownership privacy at the same time. Experimental resultsdemonstrate that the security of our scheme is not at theexpense of the performance.

For our future work, we will tryto seek ways to protect the identity privacy of data owners,which is not considered in our scheme.

REFERENCES

[1] *Amazon Web Services*, [Online]. Available: https://aws.amazon.com/cn/.

[2] D.A. Sarma, X. Dong, and A. Halevy, *Bootstrapping pay-as-you-godata integration systems*[C].ACM SIGMOD International Conferenceon Management of Data, SIGMOD 2008, Vancouver, Bc, Canada,June. DBLP, 2008:861-874.

[3] J.R. Douceur, A. Adya, W.J. Bolosky, D. Simon, and M. Theimer,*Reclaiming Space from Duplicate Files in a Serverless DistributedFile System*[C].Distributed Computing Systems, 2002.Proceedings.22nd International Conference on. IEEE, 2002: 617-624.

[4] S. Ghemawat, H. Gobioff, and S. Leung, *The Google File System*[M].SOSP '03 Proceedings of the nineteenth ACM symposium on Operatingsystems principles, 2003, 37(5): 29-43.

[5] D. Borthakur, *HDFS architecture guide*[J]. Hadoop Apache Project,2008, 53.

[6] J. Li, X. Chen, M. Li, J. Li, P.P.C. Lee, and W. Lou, *Secure Deduplication with Efficient and Reliable Convergent Key Management*[J]. IEEEtransactions on parallel and distributed systems, 2014, 25(6): 1615-1625.

[7] G.R. Blakley and C.A. Meadows, *Security of Ramp Schemes*[C].Crypto.1984, 84: 242-268.

[8] A.D. Santis and B. Masucci, *Multiple Ramp Schemes*[J]. IEEE Transactionson Information Theory, 1999, 45(5): 1720-1728.

[9] M. Wen, K. Ota, H. Li, J. Lei, C. Gu, and Z. Su, *Secure DataDeduplication with Reliable Key Management for Dynamic Updates inCpss*[J]. IEEE transactions on computational social systems, 2015,2(4): 137-147.

[10] W. Leesakul, P. Townend, and J. Xu, *Dynamic Data Deduplicationin Cloud Storage*[C]. IEEE, International Symposium on ServiceOriented System Engineering. IEEE, 2014:320-325.

[11] F. Rashid, A. Miri, and I. Woungang, *A secure data deduplicationframework for cloud environments*[C]. Tenth International Conferenceon Privacy, Security and Trust. IEEE Computer Society, 2012:81-87.

[12] M. Bellare, S. Keelveedhi, and T. Ristenpart, *Message-Locked Encryption and Secure Deduplication*[C].Annual International Conferenceon the Theory and Applications of Cryptographic Techniques.Springer, Berlin, Heidelberg, 2013: 296-312.

[13] M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, and G. Segev,*Message-Locked Encryption for Lock-Dependent Messages*[M]. Advancesin CryptologyCCRYPTO 2013. Springer, Berlin, Heidelberg,2013: 374-391.

[14] M.W. Storer, K. Greenan, D.D.E. Long, and E.L. Miller, *SecureData Deduplication*[C]. Proceedings of the 4th ACM internationalworkshop on Storage security and survivability. ACM, 2008: 1-10.

[15] P. Anderson and L. Zhang, *Fast and Secure Laptop Backups withEncrypted De-duplication*[C]. LISA. 2010.

